**Федеральное бюджетное государственное образовательное учреждение высшего образования**

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет прикладной математики, информатики и механики

Отчёт

по лабораторной работе №2

Выполнила**:** Козлова Анна Валерьевна

Студентка 2 курса направления ИТ 61 группы

Преподаватель: Трофименко Елена Владимировна

1. **Постановка задачи**

Даны два указателя: P1 на корень непустого дерева поиска и P2 на одну из вершин этого дерева, имеющих две дочерние вершины. Удалить из исходного дерева вершину P2 так, чтобы полученное дерево осталось деревом поиска. Удаление выполнять следующим образом: в левом поддереве вершины P2 найти вершину P с наибольшим значением, присвоить это наибольшее значение вершине P2, после чего удалить вершину P, (вершина P будет иметь не более одной дочерней вершины).

1. **Разработка программы**

## Модель и метод решения задачи*.*

На вход: программы подается бинарное дерево поиска(которое строится в функции BuildTree (); ) и значение вершины,которую надо удалить из дерева. В функции Search (int,node \*\*); производится поиск этой вершины в дереве. Если находится узел с таким же значением, то в функцию Delete (node \*\*,node \*\*) передается указатель на это значение и указатель на дерево. В функции после проверки на корректность полученных условий, управление передается в Delete\_1 (node \*\*, node \*\*) , где параметрами будут являться-удаляемая вершина и его левое поддерево. В левом поддереве поиска ищется максимальный элемент, значение которого будет переносится в удаляемую вершину. Максимальная вершина при этом будет удаляется.

На выход: будет выводиться измененное дерево через функцию Vyvod (node\*\*,int);

## Описание структуры данных*.*

На вход:

struct node

{

int Key; -данные которые хранятся в каждом узле

int Count; -счетчик, для одинаковых данных

node \*Left; -указатель на дочернюю вершину слева

node \*Right; --указатель на дочернюю вершину справа

};

Структура(node) является бинарным деревом поиска.

Так же на вход подается значение типа int, которое ищется в дереве и впоследствии удаляется.

На выход:выводится измененная структура

## Алгоритм решения задачи.

# Создаётся структура (node)- бинарное деревом поиска

1. Описывается класс TREE c указателями \*Tree(указатель на корень дерева),\*Res(указатель на найденную вершину), и список используемых функций для бинарного дерева.
2. В функции main() создается переменная A типа TREE для связи с классом.

Начало

1. Создается бинарное дерево через функцию A.BuildTree () и приватную фунцию void Search (int, node\*\*); .Дерево выводится на экран через A.Vyvod(A.GetTree(),0);
2. Создается цикл в котором у пользователя будет спрашиваться сколько раз произвести удаление.
3. Вводится значение el которое надо удалить из дерева.
4. Ищется указатель с таким же значением в дереве через функцию A.Poisk(el); найденный указатель передается в \*Res
5. Производится удаление вершины через функцию A.Delete(A.GetTree(),A.GetDel()); параметрами которого будут \*Tree и \*Res.
6. Если все условия были корректны вызывается приватная функция Delete\_1 (node \*\*r, node \*\*q) где \*\*r-левое поддерево(там будет искаться максимальный элемент),\*\*q-вершина, которую надо удалить
7. Когда находится максимальный элемент в левом поддереве поиска значение Key и Count переносится в удаляемую вершину. Максимальная удаляется.
8. Если все условия прошли успешно, на экран выводится измененное дерево через A.Vyvod (A.GetTree(),0);
9. **Руководство программиста**

Программа содержит 8 функций:

1. node\*\* GetTree() -возвращает указатель \*Tree на корень дерева
2. void BuildTree ()-создает бинарное дерево(вводятся данные el)
3. void Search (int, node\*\*)- по правилу бинарного дерева поиска для данных el рекурсивно ищется место в дереве(создается новый узел).
4. void Poisk (int);-поиск передаваемого значения в дереве.Указатель узела в котором Key совпадает с этим значением, записывается в \*Res
5. node\*\* GetDel()-возвращает указатель \*Res на найденную вершину
6. bool Delete (node\*\*, node\*\*) возвращает 0, если нет \*Tree, \*Res или у удаляемой вершины нет ровно двух дочерних; возвращает 1, если условия выполнились и управление передается в функцию void Delete\_1 (node\*\*,node\*\*);
7. void Delete\_1 (node\*\*,node\*\*)-функция в которой происходит удаление вершины. В левом поддереве удаляемой вершины ищется максимальный элемент, у которого не более одной дочерней вершины. Возможны 2 случая: если в левом поддереве есть только левое поддерево, то высший левый элемент будет максимальным элементом(верхний уровень). Если у нас имеется правое поддерево, то мы спускаемся по правому поддереву и ищем последний правый максимальный элемент(нижний уровень). Значения данных максимума переносится в удаляемую вершину. Максиму при этом удаляется.
8. void Vyvod (node\*\*,int)-вывод дерева на экран.
9. **Руководство пользователя**

## Пользователю нужно создать бинарное дерево поиска. По очереди вводятся с клавиатуры числа(если будут символы, или не будет введенных элементов, будет выход) до значения ‘0’.Надо стараться делать дерево с равномерной длинной его листов.

## На экран выводится исходной дерево.

## Спрашивается какой элемент надо удалить из дерева. При этом надо смотреть, чтобы удаляемый элемент присутствовал в дереве, и у элемента были 2 дочерние вершины. Иначе будет выведено окно «Ошибка». Так же нельзя вводить символ, только число.

## На экран будет выведено измененное дерево. У пользователя спрашивается продолжать ли удаление или нет. Надо вводить числа указанные в условии.

## Тестирование программы и его результаты

## Основной тест

Входные нормальные данные:

17

13 20

5 14 19 23

Удаляется число: 17

*Результат:*

14

13 20

5 19 23

## Вырожденный тест

Входные упрощённые данные:

5

3 13

Удалить число:5

*Результат:*

3

13

## Экстремальный тест

Входные упрощённые данные: (случай,когда значение элементов равно максимально допустимому значению int)

55

11 2147483647

Удаляем число:55

*Результат:*

11

2147483647

## Аварийный тест

Входные данные: (случай,когда значение элементов превышает максимально допустимое значение int):

При вводе числа больше,чем максимального значения int, дерево не создается.

*Результат:*

Программа аварийно завершается.

**Результаты работы программы и их анализ**

## Пример 1

30

11 35

5 15 31 37

2 14 19

Удалить число: 11

*Результат:*

30

5 35

2 15 31 37

14 19

## Пример 2

Дерево полученное в А.

30

5 35

2 15 31 37

14 19

Удалить число:30

*Результат:*

19

5 35

2 15 31 37

14

1. **Выводы**

Таким образом, используя деревья, я получила знания о том, как можно по-новому организовать список данных, и в частности как из этого списка можно удалить элементы, чтобы он продолжал быть упорядоченным